



# Programación Python para BigData

## Lección 2: Docker y Docker compose

## Indice

Introducción.....	3
Creación de una instancia mediante Docker.....	4
Docker.....	4

## **Introducción**

En el presente tema se ha explicado el empleo de Docker y Docker-compose para la creación de instancias ejecutables en cualquier máquina sin la necesidad de instalar nada.

En el ejercicio se pide crear tanto la guía como el servicio para la ejecución de MongoDB y PostgreSQL.

## Creación de una instancia mediante Docker

Dado que la creación de una instancia es igual para cualquiera que sea el paquete a emplear, se explicará de forma única, aclarando las diferencias cuando sean necesarias.

### Docker

A la hora de crear una instancia empleando Docker haremos lo siguiente:

1.- Accederemos a la pagina de Docker Hub y buscaremos el contenedor del paquete del que queremos crear la instancia.

MongoDB: [https://hub.docker.com/\\_/mongo](https://hub.docker.com/_/mongo)

PostgreSQL: [https://hub.docker.com/\\_/postgres](https://hub.docker.com/_/postgres)

2.- Copiaremos el comando “pull” y lo pegaremos en la consola:

MongoDB: `sudo docker pull mongo`

PostgreSQL: `sudo docker pull postgres`

Al ejecutarlo en la consola debería aparecernos lo siguiente:

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker pull postgres
[sudo] contraseña para elidas:
Using default tag: latest
latest: Pulling from library/postgres
a330b6cecb98: Pull complete
3b0b899b4747: Pull complete
cc0b2671a552: Pull complete
1a7c7505993a: Pull complete
02cdead79556: Pull complete
0d8fbe9259d6: Pull complete
974e6d476aa7: Pull complete
e9abf0d5d0bc: Pull complete
38a9de11c706: Downloading 38.01MB/73.83MB
a3864ed531fa: Download complete
de957ee6c50c: Download complete
a8eba1185eab: Download complete
67aed56271be: Download complete
```

Al finalizar aparecerá lo siguiente:

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker pull postgres
[sudo] contraseña para elidas:
Using default tag: latest
latest: Pulling from library/postgres
a330b6cecb98: Pull complete
3b0b899b4747: Pull complete
cc0b2671a552: Pull complete
1a7c7505993a: Pull complete
02cdead79556: Pull complete
0d8fbe9259d6: Pull complete
974e6d476aa7: Pull complete
e9abf0d5d0bc: Pull complete
38a9de11c706: Pull complete
a3864ed531fa: Pull complete
de957ee6c50c: Pull complete
a8eba1185eab: Pull complete
67aed56271be: Pull complete
Digest: sha256:97e5e91582e89514277912d4b7c95bceabddede3482e32395bcb40099abd9c506
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest
```

Verificamos que la creación de la imagen ha ido bien mediante

```
sudo docker image ls / sudo docker images
```

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker image ls
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
eip/fastapi         latest     240a0aaef8fc  3 hours ago  1.03GB
python              3.8        ff08f08727e5  43 hours ago  909MB
postgres            latest     346c7820a8fb  6 days ago   315MB
mongo               latest     0bcbeb494bed  9 days ago   684MB
hello-world         latest     d1165f221234  6 months ago  13.3kB
```

como se puede ver en la imagen, hay un repositorio con el nombre Mongo y otro con el nombre Postgres que son los que acabamos de crear.

3.- Ejecutamos la imagen.

Como no se ejecutan del mismo modo, los expondremos uno por uno

### Mongo

Ejecutaremos el siguiente comando:

```
docker run --name some-mongo -p 27017:27017 mongo
```

En lugar de some-mongo introduciremos el nombre que nosotros queramos darle: mongoddb

Obtendremos la siguiente salida en la consola:

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker run --name mongoddb -p 27017:27017 -d mongo
c1d572b810fef9fc1935237d2c6a0beb6d65d3218df48d749b9b64ee42e54c59
```

Verifiquemos que la imagen esta corriendo:

```
sudo docker ps
```

En mi caso por algun motivo que no he sido capaz de resolver no consigo que se ejecute la imagen. Si no me equivoco es cosa del AVX que no lo tengo activo en la maquina virtual.

### PostgreSQL

Ejecutaremos el siguiente comando:

```
sudo docker run --name postgres -p 5432:5432 -e POSTGRES_PASSWORD=123456 -d postgres
```

se pueden sustituir tanto --name como POSTGRES\_PASSWORD por los valores que nosotros decidamos, en este caso serán postgres y 123456 respectivamente

Obtendremos la siguiente salida en la consola:

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker run --name postgres -p 5432:5432 -e POSTGRES_PASSWORD=123456 -d postgres
e17db40ceec69401d1c14523656304d1a9011ecc4a184f775c168472d337cb1c
```

Verifiquemos que la imagen esta corriendo:

```
sudo docker ps
```

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacion_de_contenedores/Actividad_2$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS                               NAMES
e17db40ceec6   postgres  "docker-entrypoint.s..."  2 minutes ago  Up 2 minutes  0.0.0.0:5432->5432/tcp, :::5432->5432/tcp  postgres
```

Una vez obtenemos este resultado la instancia esta corriendo y se quedará corriendo en segundo plano a no ser que la detengamos.

Si deseamos detener la ejecución:

```
sudo docker kill name
```

Siendo name el nombre de la instancia a detener.

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacon_de_contene
dores/Actividad_2$ sudo docker kill postgres
postgres
```

## Docker - compose

En este caso dependemos de un archivo .YML que llamaremos mongo.yml en el caso de Mongo DB y postgres.yml en el caso de PostgreSQL

En ambos casos se arranca del mismo modo, ejecutando

```
sudo docker-compose -f file.yml up
```

En el caso de MongoDB sustituiremos name por mongo y en el caso de PostgreSQL por postgres y obtendremos una salida parecida a la siguiente:

```
(BigData) elidas@FireBall:~/Escritorio/Master/8-Programacion_para_BigData/Leccion_2-Creacon_de_contene
dores/Actividad_2/PostgreSQL$ sudo docker-compose -f stack.yml up
Pulling adminer (adminer:)...
latest: Pulling from library/adminer
a0d0a0d46f8b: Pull complete
153eea49496a: Pull complete
11efd0df1fcb: Pull complete
b3f3214c344d: Pull complete
6b773013e8ac: Pull complete
dc6d797d83c8: Pull complete
66d7e4d212e9: Pull complete
071d4aa8c63e: Pull complete
e917640f6ebf: Pull complete
a6ede87dcb0f: Pull complete
b5191adc889b: Pull complete
73f76d757bfb: Pull complete
6a3a39211c2d: Pull complete
e91e9f0991f5: Pull complete
e11751b0fcc: Pull complete
Digest: sha256:9d187c3025d03c033dcc71e3a284fee53be88cc4c0356a19242758bc80cab673
Status: Downloaded newer image for adminer:latest
Creating postgresql_db_1 ... done
Creating postgresql_adminer_1 ... done
```

Continuará apareciendo código unos segundos y una vez se detenga, habrá finalizado el inicio y podremos usar la instancia.

Para detener la ejecución pulsaremos Ctrl + C y esperaremos a que finalice los procesos de detención.